Parseline – Bibliotecă C++ pentru interpretarea și gestionarea comenzilor CLI/REPL

1. Scopul proiectului

Scopul acestui proiect este dezvoltarea **Parseline**, o bibliotecă scrisă în C++ pentru interpretarea și gestionarea comenzilor din linia de comandă (CLI) și în regim REPL (Read–Eval–Print Loop). Biblioteca va oferi o interfață clară și flexibilă pentru definirea comenzilor, parsarea argumentelor și execuția controlată a funcțiilor asociate acestora, atât în mod static (argv) cât și interactiv.

---

2. Motivația dezvoltării propriei soluții

Deși în ecosistemul C++ există soluții populare precum `cxxopts`, `CLI11`, `TCLAP` sau `Boost.Program_options`, ele tind să fie:

- **Supradimensionate pentru aplicații simple** – necesită boilerplate semnificativ chiar și pentru cazuri de bază.
- **Greu de personalizat** – personalizarea comportamentului REPL sau a fluxurilor de output poate fi limitată sau non trivială.
- **Focalizate doar pe parsing CLI** – fără suport nativ pentru REPL sau interacțiune în buclă.
- **Nealiniate cu unele practici moderne** – folosirea masivă a template-urilor, dependințe externe sau lipsa modularității.

Prin dezvoltarea Parseline, se urmărește:

- Crearea unei soluții **flexibile, extensibile și lightweight**, potrivite pentru aplicații embedded, instrumente de linie de comandă sau sisteme REPL integrate.
- Înțelegerea în profunzime a mecanismelor de interpretare a comenzilor și a infrastructurii de tip shell/terminal.

- Crearea unui nucleu **reutilizabil** pentru proiecte personale sau comerciale unde e necesară interacțiunea textuală cu utilizatorul.

---

3. Obiective generale

- Crearea unei biblioteci scrise în C++ modern, fără dependențe externe, compatibilă cu standardul C++17 sau superior.
- Permite definirea de comenzi cu nume, descrieri, argumente și funcții asociate.
- Oferirea unui mod interactiv (REPL) cu buclă proprie de execuție.
- Asigurarea unui mod simplu de integrare atât în aplicații de linie de comandă, cât și în interfețe embedded sau simulatoare.
- Sprijinirea testarii și a redirecționării ușoare a intrării și ieșirii (custom output streams).

---

4. Cerințe funcționale

Biblioteca Parseline va trebui să îndeplinească următoarele cerințe funcționale:

1. **Înregistrarea comenzilor**:
    - Comenzile pot fi înregistrate dinamic, fiecare cu:
        - Nume textual
        - Funcție asociată
        - Descriere pentru help
        - Set de argumente opționale
2. **Mod CLI static**:
    - Acceptă input din `int argc, char* argv[]` și execută funcția asociată comenzilor recunoscute.
    - Sistem de help automat și validare a argumentelor.

3. **Mod REPL**:
    - Execuție interactivă în buclă (`while (true)`) cu suport pentru:
        - Comenzi multiple
        - Ieșire din REPL printr-o comandă (`exit`, `quit`, etc.)
        - Afișarea comenzilor disponibile și autocompletare opțională (extensibil)
4. **Personalizarea fluxurilor de I/O**:
    - Permite injectarea unor fluxuri personalizate (`std::istream`, `std::ostream`) pentru input și output (ex: pentru testare automată, interfețe grafice text based sau embedded I/O).
5. **Gestionare erori**:
    - Erorile de parsing și execuție vor fi capturate și afișate într-un mod controlat și informativ.
6. **Extensibilitate**:
    - Ușor de extins pentru a adăuga:
        - Argumente poziționale și named flags
        - Aliasuri pentru comenzi
        - Comenzi ascunse pentru moduri debug/admin

---

5. Cerințe nefuncționale

- **Performanță**: Biblioteca trebuie să aibă un timp de inițializare și răspuns scăzut, chiar și cu un număr mare de comenzi.
- **Portabilitate**: Codul trebuie să funcționeze pe majoritatea platformelor (Windows, Linux, macOS).
- **Modularitate**: Funcționalitățile trebuie împărțite în componente distincte (parser, command handler, REPL).

- **Simplitate în utilizare**: Codul client trebuie să fie ușor de scris și citit.
- **Documentație**: Fiecare funcție publică trebuie să fie documentată clar, iar biblioteca să includă un ghid de utilizare.

---

6. Public țintă și domenii de aplicabilitate

Biblioteca Parseline este adresată:

- Dezvoltatorilor de instrumente CLI cross-platform
- Cercetătorilor care au nevoie de un mediu REPL ușor de integrat în simulatoare sau aplicații științifice
- Studenților și entuziaștilor care doresc să înțeleagă arhitectura internă a unei interfețe REPL/CLI
- Proiectelor embedded unde interacțiunea se face prin UART sau terminal serial